Shedding Our Skins?
Reflections on a liaison librarian’s attempt to learn how to scrape data from the web using Python

Jeremy Darrington
Politics Librarian, Princeton University Library
Presentation for the IASSIST 2014 Conference (Toronto)
The “what”

- “Web scraping”
  - Using software to extract info from websites
- “Python”
  - A popular programming language that is powerful and flexible, but also easy for beginners to learn
- “Liaison librarian”
  - Someone hired for his/her subject expertise—not technological prowess!—to support research in an academic department
The “why”

- Changing research environment for social scientists
  - New sources of data on the web
  - Many old sources now digitized or only published online
- Possibility (expectation?) of new roles and new skills for liaison librarians
  - Understanding researchers’ environment and tools
  - Online data collection support/advice?
  - Web scraping as a new form of collection development?
- Personal interest
  - Programming is fun! (no, I’m not joking)
  - The tech support when tech support isn’t available
The “catalyst”, part I

- “Web Scraping and Text Processing with Python” workshop in January 2014
Web Scraping and Text Processing with Python Workshop, January 27-31, 2014

Dates: January 27 - 31, 2014
Morning Session: 9:30–11:30am
Afternoon Session: 1:30–3:30pm
Location: TBA
Instructor: Radhika Saksena

**Registration** The registration for this workshop is now closed due to space limitations. If you are interested in being added to the waitlist, please fill out the waitlist form.

*Note: this workshop is open only to Princeton affiliates*

Over the last decade, both the variety and amount of data available to social scientists have expanded. These new data sources include administrative records (e.g., voter files, campaign finance and lobbying records), geo-referenced data (e.g., satellite maps, geocoded event data), and texts (e.g., speeches, court rulings, legislative bills). Many of these data sources can be accessed through the World Wide Web and as a consequence, techniques such as web scraping have become an essential part of social scientists’ toolkit. The objective of this workshop is to introduce basic tools and techniques for automatic content extraction, parsing and other data-handling tasks that are commonly encountered in data-intensive research projects. The course will be taught in Python, and only a basic knowledge of general computing and programming (such as the R statistical programming taught at the Introductory Statistical Programming Camp) is assumed. We will cover techniques ranging from Python regular expressions and file manipulation, to the popular web scraping library "Beautiful Soup" and PDF content extraction. The course ends with an introduction to the Twitter API for accessing Twitter content.
The “catalyst”, part II

- This presentation!
  - There’s nothing like a deadline and the risk of professional embarrassment to help you reach your goals!
The “how”, part I

- Codecademy course on Python
  - [http://www.codecademy.com/tracks/python](http://www.codecademy.com/tracks/python)
**Something of Value**

For paperwork and accounting purposes, let's record the total value of your inventory. It's nice to know what we're worth!

1. Create a variable called `total` and set it to zero.
2. Loop through the `prices` dictionaries.
3. For each key in `prices`, multiply the number in `prices` by the number in `stock`. Print that value into the console and then add it to `total`.
4. Finally, outside your loop, print `total`.

⚠️ **Stuck? Get a hint!**

⚠️ **Oops, try again.** It looks like your code did not print the correct total.
The “how”, part II

- Setting up Python on my own computer
But this could clearly be rewritten more parsimoniously, which not only reduces lines of code but also extends it to work with additional URLs and search strings. It took me a few tries, but the script below will now return results for additional URLs and search strings.

```python
from bs4 import BeautifulSoup


soups = []
for url in urls:
    response = urllib2.urlopen(url).read()
    soup = BeautifulSoup(response)
    soups.append(soup)

searchStrings = ["はなく", "景気回復", "経済", "TPP", "震災", "安全保障"]

patterns = []
for searchString in searchStrings:
    pattern = re.compile(searchString)
    patterns.append(pattern)

for i, soup in enumerate(soups):
    total = 0
    for j, pattern in enumerate(patterns):
        # Use the index from this loop to get each search string and the index from the parent loop to feed the proper URL
        print '%s occurs on %s %d times.' % (searchStrings[j], urls[i], len(pattern.findall(soup.text))
        total += len(pattern.findall(soup.text)

print 'Total occurrences of all search terms is %d' % total
```

Total occurrences of all search terms is 20

Total occurrences of all search terms is 15
The “how”, part II

- Viewing/sharing notebooks
  - http://nbviewer.ipython.org/
  - http://www.wakari.io/
  - My example notebooks: https://www.wakari.io/jdarring
Reflections

- Scraping is a valuable tool, but legal and ethical considerations abound
- Lots of great tools and tutorials, but …
- Learning any new skill takes time and repetition
  - The best learning happens when you tackle real problems
  - Group learning can be helpful, esp. in programming
  - Library administrators should be supportive and actively encourage exploration
- Programming skills are useful and transferable
Questions?
jdarring@princeton.edu